

## STREAM SWITCH FABRIC

### FIELD OF THE INVENTION

5           The present invention relates generally to the field of stream data and, more specifically, to processing of stream data within a stream switch fabric.

### BACKGROUND OF THE INVENTION

10           A packet switched network is one in which data is communicated as discrete packets, each packet containing a header. The header contains information regarding the packet's source, ultimate destination, etc. For instance, in the case of a Transmission Control Protocol (TCP)/Internet Protocol(IP) packet, the header includes the IP source address, TCP source port,  
15           the IP destination address and the TCP destination port. Each packet is individually transmitted to its ultimate destination through a series of network switches (bridges, IP switches, IP routers, layer 4 load balancers, etc.). Most network switches examine the header to determine the packet's ultimate destination, and the network switch decides how to forward the packet based on  
20           this header information. A network switch that operates in this manner remains "ignorant" of the packet content since the actual content of the packet is irrelevant in making forwarding decisions.

            Recently, network switches (e.g. bridges, IP switches, IP routers, layer 4  
25           load balancers, etc.) have emerged, which are "content aware". These types of switches are often referred to as "content switches". Content switches are embedded with specific knowledge of, and monitor or participate in, the upper layer protocols (e.g., HTTP etc.). Content switches examine and sometimes modify the packets of data transmitted through the network. These content  
30           switches are typically designed around a packet-oriented switch fabric similar to the way bridges, IP routers and layer 4 load balancers are designed. In some

instances, content switches may also perform bridging, routing and load balancing functions.

Conventional content switches have been made content aware only for applications that are popular and that will benefit from content aware switching or processing. These applications have been and likely will continue to be stream-oriented applications. A stream is a series of bytes transferred from one application to another and stream-oriented applications create or manipulate data as streams of bytes. In the case of a stream being transmitted within a packet switched network, the stream of data is divided and transmitted via a flow of packets, a flow of packets being a bidirectional set of packets going between two end points within two end stations. At the application end points, streams of bytes are passed through layer 4 processing, such as a TCP layer. On behalf of the sending application, the layer 4 processing divides the data stream arbitrarily into segments, applies a segment header to identify the stream and the relative order of the data, and applies a layer 3 header such as an IP header. On behalf of the receiving application, the layer 4 processing re-orders mis-ordered segments, requests retransmission of lost packets and reconstitutes the byte stream. It is noted that it is possible for packet switched networks to mis-order and/or lose packets.

One problem with the current design of content switches is that as content switching and content modification become more sophisticated, the ability to perform functions on a packet by packet basis will become awkward, inefficient and virtually impossible. Thus, there will be the need for the content switch to reconstitute the original data stream and re-segment it, this functionality being referred to as a layer 4 proxy function (for example a TCP proxy function).

As content functions become more sophisticated and processing intensive, a content switch will need to distribute the processing among

processing elements within the switch to become faster and more efficient. These processing elements may be identical and simply load share the work or they may be optimized for certain applications or portions of applications. Connecting these processing elements with a packet fabric will require that the  
5 layer 4 proxy function be performed by or next to each processing element.

The location of the layer 4 proxy function next to each processing element, however, is less than optimal because some applications may only determine the most appropriate processing entity by examining the content and  
10 because some applications may require multiple processing elements to perform content functions on a data stream in series. In these cases, the proxy function would be done more times than necessary and the stream would be transmitted more times than necessary.

15 It would thus be beneficial to provide a fabric, which would manage/switch data streams, allowing the layer 4 proxy function to be more appropriately placed and performed less often.

#### SUMMARY OF THE INVENTION

20 The present invention is directed to apparatus and method for processing data streams within a stream switch fabric. In embodiments of a stream switch fabric according to the present invention, at least one stream queue is coupled to a stream queue controller. A producer operates to output properly ordered  
25 portions of a data stream, hereinafter referred to as substreams, to the stream queue. Subsequently, the stream queue controller operates to control the outputting of a portion of the data within the stream queue to a consumer of the stream queue. In embodiments of the present invention, the stream queue controller can initiate further actions within the stream queue such as copying a  
30 portion of the data within the stream queue to a content processing element for analysis; redirecting the data within the stream queue to another consumer of

the data stream, such as another processing element or an interface with a packet switched network; modifying a portion of the data within the stream queue; and/or transferring a portion of the data within the stream queue to another stream queue within the stream switch fabric. In some embodiments of the present invention, the stream queue is controlled by control signals being received from an external element such as the consumer of the stream queue.

According to a first broad aspect, the present invention is a stream switch fabric including at least one stream queue and a stream queue controller coupled to the at least one stream queue. The stream queue operates to receive and store a plurality of properly ordered substreams of a data stream from a producer of the data stream and the stream queue controller operates to control outputting of at least a portion of the data within the at least one stream queue to a consumer of the stream queue.

In some embodiments of the present invention, the stream queue controller further operates to receive a control signal associated with the stream queue. In these cases, the control signal could be an indication of at least one consumer attribute for the at least one stream queue, such as the actual consumer that is assigned as the consumer of the stream queue or the number of bytes of the data within the stream queue that are to be output to the consumer of the stream queue. Further, the control signal could be an instruction to trigger copying of at least a portion of the data within the stream queue to the consumer of the stream queue, an instruction to trigger forwarding of at least a portion of the data within the stream queue to the consumer of the stream queue and deleting of this portion of the data within the stream queue, or another such trigger instruction.

According to a second broad aspect, the present invention is a stream switch fabric including reception means for receiving a plurality of properly ordered substreams of a data stream from a producer of the data stream; storage

means for storing the substreams; and control means for controlling outputting of at least a portion of the data within the means for storing the substreams to a consumer of the data stream.

5 According to a third broad aspect, the present invention is a method of processing streams of data. The method includes receiving properly ordered substreams of a data stream; storing the substreams within a stream queue associated with the data stream; and outputting at least a portion of the data within the stream queue to a consumer of the stream queue.

10 Other aspects and advantageous features of the present invention will become apparent to those ordinarily skilled in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying figures.

15 BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will be more clearly understood by reference to the following detailed description of an exemplary embodiment in conjunction with the accompanying drawings, in which:

20 Figure 1 illustrates a flow chart of a set of steps performed by an initiator of a new stream queue according to the present invention;

25 Figure 2 illustrates a simplified block diagram of a stream processing node according to an embodiment of the present invention;

Figure 3 illustrates a flow chart of a set of broad steps performed within a stream processing node according to an embodiment of the present invention;

30 Figure 4 illustrates a simplified block diagram of a stream processing

node with an I/O element coupled to a packet switched network according to an embodiment of the present invention;

5 Figure 5 illustrates a logical block diagram of a sample implementation of the stream processing node of Figure 4 in a packet switched network;

Figure 6 illustrates a detailed block diagram of the stream processing node of Figure 4 according to an embodiment of the present invention;

10 Figure 7 illustrates the block diagram of Figure 6 with the signalling depicted for one particular sample operation;

15 Figure 8 illustrates a flow chart of a set of steps performed by an Input/Output (I/O) element as a producer for a stream queue according to an embodiment of the present invention;

20 Figure 9 illustrates a flow chart of a set of steps performed by a content processing element within a stream processing node according to an embodiment of the present invention;

Figure 10 illustrates a flow chart of a set of steps performed by an I/O element as a consumer for a stream queue according to an embodiment of the present invention;

25 Figure 11 illustrates the block diagram of Figure 6 with the signalling depicted for another particular sample operation;

30 Figure 12 illustrates a flow chart of a set of steps performed by a decryption processing element within a stream processing node according to the present invention; and

Figure 13 illustrates a block diagram of an alternative stream processing node to that illustrated in Figure 6.

#### DETAILED DESCRIPTION OF THE INVENTION

5

The present invention provides a method and apparatus for content based decision making for forwarding/processing data streams. Within embodiments of the present invention, properly ordered portions of data streams, hereinafter referred to as substreams, are input to stream queues within a stream fabric; stream queues being data structures that store continuous subsets of data streams while they await processing or transmitting. Each stream queue has an assigned producer and an assigned consumer, each of which have set controls over the stream queue. For instance, in one embodiment, the producer of the stream queue can push substreams and/or control information onto the stream queue while the consumer of the stream queue can request to pop portions of the data from the stream queue to the consumer, to copy portions of the data from the stream queue to the consumer, to transfer portions of data from the stream queue to another stream queue, and to push data and/or control information onto the stream queue. Further, the consumer can request the stream fabric to adjust the consumer attributes such as to change the consumer that is to control the stream queue and/or change the number of bytes transferred to the consumer during a pop/copy. The changing of the actual consumer is hereinafter referred to as rescheduling of the stream queue.

25

Figure 1 illustrates a flow chart of a set of steps performed by an initiator of a new stream queue according to an embodiment of the present invention. A new stream queue might need to be initiated in the case that a new stream of data enters the stream fabric, such as a new stream from a packet switched network. In one embodiment, the initial rights of the producer and consumer are set to the initiator of the new stream queue as depicted in step 102. Next, the

30

initiator of the new stream queue sets the priority of the stream queue relative to other stream queues within the stream fabric at step 104 and sets the criteria for the first pop/copy of data from the stream queue at step 106. Finally, at step 108, the initiator of the stream queue sets the initial consumer of the stream queue at step 106 by requesting a change in the consumer attributes for the stream queue, thus rescheduling the stream queue. This first rescheduling of the stream queue in many embodiments is done to direct the first copy of the data within the stream queue to a default processor that will always receive the right to consume the stream, such as a content processing element. In other embodiments, the initial consumer of the stream queue could be determined on a round robin basis, or on some other basis. Further, the initial consumer could be a set of possible consumers wherein the least burdened of the set, or the consumer in the set with the highest priority receives the initial right to consume the stream queue. Those skilled in the art will recognize that there are many variations for determining the initial consumer of the stream queue and that each of these variations will fall within the scope of the invention. It should further be understood that the above steps for initiating a stream queue could be performed in a different order.

Figure 2 illustrates a simplified block diagram of a stream processing node according to an embodiment of the present invention. As depicted, stream processing node 50 comprises a producer of a stream of data 60 (hereinafter referred to as a producer), a content processing element 80, a consumer of a stream of data 90 (hereinafter referred to as a consumer) and a stream fabric 70 coupled between the other elements 60,80,90. Each of the producer and consumer 60,90 could be an Input/Output (I/O) element, a processing element, a computer, a memory device, a storage device, a cache device or another device that can produce and/or consume a data stream.

The stream fabric 70, as depicted in Figure 1, comprises a stream queue controller 72 and a set of stream queue(s) 74; the stream queue(s) 74, as



described above, being data structures that store continuous subsets of data streams while they await processing or transmitting. The stream queue(s) 74 may be one or more buffers, registers and/or any other type of temporary storage devices. The stream queue(s) 74 may be hierarchical. The stream queue controller 72 controls the stream queue(s) 74 as will be described in detail herein below. Although only one stream queue controller 72 is illustrated for ease of explanation, it should be recognized that more than one could be implemented within the stream fabric 70. The stream queue controller 72 may be an application specific integrated circuit (ASIC), a reduced instruction set computer (RISC) processor, a complex instruction set computer (CISC) processor, a digital signal processor (DSP) or another processing element.

Although component 60 is labelled as a producer of a stream of data and component 90 is labelled as a consumer of a stream of data, it should be understood that the producer 60 could also operate as a consumer of a stream of data and the consumer 90 could operate as a producer of a stream of data. This is illustrated in detailed examples described herein below in which producer 60 is an Input/Output (I/O) element and the consumer 90 is a processing element. Further, it should be understood that the content processing element could be considered an initial consumer of streams of data. Yet further, although only a single producer 60, a single consumer 90 and a single content processing element 80 is shown, a plurality of one or more of these elements could be implemented in a stream processing node according to the present invention.

Figure 3 illustrates a flow chart of the broad steps performed within the stream processing node 50 according to an embodiment of the present invention. Firstly, at step 200, the producer 60 pushes properly ordered portions of a stream of data (substreams) to a stream queue 74 within the stream fabric 70. The producer 60 could have acquired the stream of data from many different sources including an internal memory source, an internal processing element and/or an external component or network. For instance, in the case

that the producer is an I/O element, the producer 60 might have received a stream of data (voice, data, video, etc.) segmented within a series of data packets from a packet switched network. In this case, the producer 60 would classify the received packets into separate streams, reconstitute substreams of the data within the packets and push the reconstituted substreams to a stream queue 74 within the stream fabric 70.

Next, as depicted at step 300 of Figure 3, a copy of the leading portion of the data within the stream queue 74 is forwarded to the content processing element 80. This is followed at step 400 by the content processing element 80 processing the content in the leading portion of the data within the stream queue and, based upon this content, deciding what to do with the particular stream. For instance, as described below in more detail, the content processing element 80 could decide to: consume the stream (i.e. pop and process all data received at the stream queue), consume a portion of the stream (i.e. pop and process a portion of the data received at the stream queue), add data to the stream (i.e. push data to the stream queue), delete data from the stream (i.e. simply pop data from the stream queue and disregard it), change data in the stream (i.e. pop data from the stream queue and push modified data back onto the stream queue), redirect the right to consume the stream to another potential consumer (i.e. change the consumer attributes for the stream queue), request more data from the stream queue 74 (i.e. copy a further portion of the data from the stream queue), transfer data from the stream queue 74 to another stream queue (i.e. pop a portion of the data on the stream fabric and push this data onto another stream queue) etc. The content processing element 80 then sends instructions to the stream queue controller 72 indicating the decision that was made. For instance, if the content processing element 80 decides to consume the stream then it instructs the stream queue controller 72 to push portions of the data in the stream queue 74 to that processing element. If the content processing element 80 decides to reroute the stream, it instructs the stream queue controller 72 which consumer should have the right to consume the data within the

particular stream queue 74.

Subsequently, as depicted at step 500, the stream queue controller 72 performs the copy, pop, push, reschedule and/or transfer operation on the stream queue 74 as requested, each of these functions described by example in more detail herein below. Finally, if the stream queue controller 72 reschedules the stream queue to another consumer, such as the consumer 90 of Figure 1, or essentially continues to assign itself as the consumer by selecting to pop data from the stream queue, the data within the stream queue is processed by the consumer as required at step 600. At this point, the results from the processing of the data stream may be output from the stream processing node, saved within the stream processing node or output back within a second stream queue within the stream fabric 70. In this last case, the consumer is essentially the producer of the second stream queue and possibly the initiator of this second stream queue.

Figure 4 illustrates a simplified block diagram of a stream processing node with an I/O element coupled to a packet switched network according to an embodiment of the present invention. In this embodiment, the producer 60 of Figure 2 is an I/O element which receives data packet flows from a packet switched network and outputs properly ordered substreams of the data streams to respective stream queues within the stream fabric 70. The processing element 90 could be utilized as a consumer of the data streams if it is deemed necessary to perform processing on any of the data within the stream queues of the stream fabric 70. Further, as will be described herein below in detail, the I/O element 60 could be utilized as a consumer of data streams if any of the data streams must be routed back onto the packet switched network and the processing elements 80,90 could be utilized as producers of data streams if, after processing a data stream or a portion of a data stream, substreams of a new or modified data stream are output to a new stream queue within the stream fabric 70.

It should be understood that, although Figure 4 shows only one I/O

element 60, one content processing element 80 and one processing element 90, it should be understood that there may be any number of I/O elements, content processing elements, processing elements and/or other producer and consumer components within the stream processing node.

5

Figure 5 illustrates a logical block diagram of a sample implementation of the stream processing node 50 of Figure 4 in a packet switched network. As depicted, a first client 120 is communicating with a second client 130 via the stream processing node 50 utilizing a layer 4 protocol such as TCP. In this case, the first client 120 is coupled to the packet switched network via a layer 4 socket 122 and a layer 4 port 124 and the second client 130 is coupled to the packet switched network via a layer 4 socket 132 and a layer 4 port 134. The I/O element 60 within the stream processing node 50, as depicted in Figure 4, is logically comprised of a layer 4 socket 142 coupled to the stream fabric 70 and a layer 4 port 144 coupled between the layer 4 socket 142 and the packet switched network.

In this case, a stream of data output from the first client 120 is segmented by the layer 4 socket 122 and the segments of the stream are output as data packets with layer 3 and layer 4 headers (such as TCP and IP headers) from the layer 4 port 124. The layer 4 port 144 along with the layer 4 socket 142 operate to receive the data packets transmitted from the layer 4 port 124 and reconstitute the data stream output from the first client 120. The reconstitution of the data stream is done with well-known layer 4 protocol techniques including removing the packet overheads from the packets, re-ordering the data within the packets into the correct order and requesting the retransmission of any packets that were lost. The layer 4 socket 142 then forwards substreams of the data stream to the stream fabric 70. In the case that the stream processing node 50 determines that the data stream is to be forwarded to the second client (possibly due to an HTTP address within the content of the data stream), the stream fabric 70 outputs portions of the stream to the layer 4 socket 142 which along with the

layer 4 port 144, segments the stream data and outputs packets including the segmented data and layer 3 and layer 4 headers to the layer 4 port 134. Similar to the above described operation for the layer 4 port 144 and layer 4 socket 142, the layer 4 port 134 and layer 4 socket 132 proceed to receive the packets of the data stream, reconstitute the data stream and output the data stream to the second client 130.

It should be recognized that although the above description is specific to communications from the first client 120 to the second client 130, communications could further be directed in the opposite direction or other clients could be included within the system. Further, it should be understood that the bidirectional communications between the layer 4 ports 124,144 could be considered a flow of packets while the bidirectional communications between the layer 4 ports 144,134 could further be considered a flow of packets.

Figure 6 illustrates a detailed block diagram of the stream processing node 50 of Figure 4 according to an embodiment of the present invention. This implementation of the stream processing node 50 could be utilized within the example described above with reference to Figure 5. As depicted, the stream processing node of Figure 6 comprises I/O element 60, stream fabric 70, content processing element 80 and processing elements 90,100 which correspond to first and second application processing elements respectively. It should be understood that other I/O elements and/or processing elements could also be included within the stream processing node 50 of Figure 6. Further, although the stream fabric 70 is depicted with four stream queues 74, other numbers of stream queues could be implemented.

As illustrated in Figure 6, the I/O element 60 comprises a processor 62 that includes a flow classification block 64, a layer 4 termination block 66 and a layer 4 segmentation block 68 which together are logically equivalent to the layer 4 port 144 and layer 4 socket 142 of Figure 5. The flow classification block

62 classifies packets received from a packet switched network into separate flows of packets based upon the flow identifiers attached to the packets, the flow identifiers including the source layer 3 address, the source layer 4 port, the destination layer 3 address, the destination layer 4 port and the layer 4 identifier (i.e. UDP or TCP etc.). Each flow of packets corresponds to a different stream of data. The layer 4 termination block 66 performs well-known reconstituting of the data stream by removing the packet overhead from the data, re-ordering the data within the received packets and requesting re-transmissions of packets that have been lost. The output from the layer 4 termination block 66 are substreams including correctly ordered portions of the overall data stream. The layer 4 segmentation block 68 performs well-known segmentation operations of incoming data from the stream fabric 70 and outputs packets including the segmented data along with layer 3 and layer 4 headers to a packet switched network.

The operation of the stream processing node 50 of Figure 6 will now be described for one particular scenario with reference to Figures 7 through 10. In this scenario, a data stream arrives at the I/O element 60 and has an indication within its content of where the data stream should be ultimately forwarded. For example, the content could include an HTTP address that indicates the appropriate location for the stream processing node to route the particular data stream. Figure 7 illustrates the block diagram of Figure 6 with the signalling depicted for this scenario.

As illustrated in Figure 7, a flow of data packets, as indicated by signal 702, is received at the I/O element 60 and a series of substreams of the data stream contained within the received data packets is subsequently output to a stream queue (in this case SQ2) within the stream fabric 70 as indicated by signal 704. In this case, the I/O element 60 is the producer of the stream queue SQ2.

Figure 8 illustrates a flow chart of a set of steps performed by an I/O

element as a producer for a stream queue according to an embodiment of the present invention. Firstly, the I/O element receives a flow of data packets at step 202. These packets may represent a single segmented data stream from a single sender, many segmented data streams from many senders, or many segmented data streams from a single sender. For example, an I/O element may receive 100 IP packets which represent a single data stream made up of 100 packets, or multiple data streams each made up of fewer than 100 packets.

Next, at step 204, the I/O element reads the flow identifiers of the packets within the flow at step 204. As described above, the flow identifiers include the packets' source layer 3 address, source layer 4 port, destination layer 3 address, destination layer 4 port and a layer 4 identifier (i.e. UDP or TCP etc.). Each flow identifier is unique to a particular data stream. Therefore, the I/O element can group the received data packets based upon the data stream which they are associated with by grouping the packets with identical flow identifiers.

Next, the I/O element terminates the layer 4 protocol for the packets using well-known techniques at step 206. This termination of the layer 4 protocol includes removing packet overhead such as the layer 3 and layer 4 headers from the data, re-ordering the data into the proper order as necessary and requesting retransmission of any packets that are found to be lost or delayed beyond a threshold level. The result of the termination of the layer 4 protocol is the reconstitution of data streams or at least portions of the data streams within the I/O element.

Finally, at step 208, the I/O element pushes properly ordered substreams of data bytes to corresponding stream queues. If no stream queue has yet been assigned for a particular received stream, the I/O element preferably goes through an initiation procedure such as that described above with reference to Figure 1.

Now returning to Figure 7, the substreams 704 output from the I/O element 60 are received at a stream queue 74 (SQ2 in Figure 7). Once the data stream is at least partially received by the stream queue 74, a copy of a portion of the data within the stream queue is forwarded to the initial consumer of the stream queue as indicated by the signal 706, the initial consumer being the content processing element 80 in the scenario of Figure 7. The stream queue controller 72 then waits for instructions from the content processing element 80.

Figure 9 illustrates a flow chart of a set of steps performed by a content processing element within a stream processing node according to an embodiment of the present invention. Firstly, the content processing element receives a portion of a data stream at step 402. Next, it determines if the portion is sufficient to analyse the stream at step 404. It might not be a sufficient portion of the stream in many instances such as in the case that the content processing element has not yet received a complete HTTP address to indicate proper routing of the data stream. If there is not a sufficient portion of the data stream to analyse, the content processing element requests, at step 406, for an additional copy operation to be performed such that an additional portion of the data within the stream queue is copied to the content processing element. Next, the content processing element processes the content within the data that it has received at step 408. Three possible conclusions of many from the analysis could be that a URL is detected as depicted at step 410, an intermittent processing is determined to be required at step 422 and encrypted data is detected at step 426.

If a URL is detected at step 410, the content processing element determines if a URL translation is required at step 412. In some embodiments of the present invention, the content processing element includes a URL translation table. If a URL is detected that is included within the table, a translation of the URL is required. One skilled in the art could conceive of other cases in which a translation in the URL would be required as well.



If no translation is required, the content processing element pushes a flow context ID to the stream queue at step 414. The flow context ID is a number that indicates to an I/O element that later processes the data stream which flow  
5 within the packet switched network to output the data stream. In the case that no previous data stream has been assigned the particular flow that the content processing element determines is necessary to properly route the data stream, the content processing element preferably pushes a flow context ID that includes the required layer 3 destination address and the layer 4 destination port for the  
10 data stream. With this information, an I/O element can produce a flow identifier for the packets of that required flow which can preferably later be accessed with use of an assigned flow context ID number. Finally, after pushing the flow context ID to the stream queue, the content processing element, at step 416, requests rescheduling of the stream queue to the proper I/O element.

15 If at step 412 it is determined that the URL requires a translation, the content processing element requests a pop of the original URL data from the stream queue at step 418. Once the content processing element receives the pop of the original URL, the content processing element pushes, at step 420, the  
20 translated URL to the stream queue to replace the original URL. Once completed, the content processing element continues to proceed with the steps 414 and 416 described herein above.

25 If an intermittent processing is required as detected at step 422, the content processing element requests a transfer of a set number of bytes from the first stream queue to a second stream queue at step 424. This intermittent processing could be required in cases that the content processing element can determine how to properly proceed with only a portion of the data stream or if the content processing element determines that a change within the data stream  
30 is required intermittently. This could occur for example in cases that a URL must be translated periodically during a single data stream. Once, the set

number of bytes are transferred, the content processing element returns to step 402 of the flow chart of Figure 9.

If encrypted data is detected at step 426, the content processing element requests rescheduling of the stream queue to a decryption engine, such as an SSL engine, at step 428.

The particular scenario being depicted within Figure 7 is consistent with the content processing element detecting a URL within the data stream at step 410 and proceeding through steps 412, 414 and 416 of Figure 9. Therefore, as depicted in Figure 7, the result of the content processing element 80 detecting a URL within the data stream is the sending of a request for rescheduling of the stream queue SQ2 to the I/O element 60 and an indication for a pop to be initiated to the I/O element 60, as indicated by the signal 708 between the content processing element 80 and the stream queue controller 72. The result of this request is the stream queue controller changing the consumer attributes associated with the stream queue SQ2 such that the consumer rights belong to the I/O element 60. In this particular case, the producer and consumer of the stream queue SQ2 are both the I/O element 60, though it should be understood that the content processing element could have requested the rescheduling of the stream queue to another I/O element. Next, the stream queue controller 72 initiates a pop of data from the stream queue SQ2 to the new consumer, I/O element 60 as indicated by signal 710, and the I/O element 60 consumes the data within the stream queue SQ2.

25

Figure 10 illustrates a flow chart of a set of steps performed by an I/O element as a consumer of a data stream according to an embodiment of the present invention. Initially, the I/O element receives a pop of a portion of a stream from a stream queue at step 602 and analyses the flow context ID provided by the content processing element to identify a flow identifier for the particular stream at step 604. If the flow context ID is simply an assigned

30

number for a particular flow, the I/O element looks up the flow identifier from a flow identifier table. If the flow context ID includes a layer 3 destination address and layer 4 destination port, the I/O element generates a new flow identifier with this information along with known information with respect to  
5 the layer 4 ports that are available within the I/O element.

Next, the I/O element segments the stream based upon a layer 4 protocol at step 606, attaches layer 3 and layer 4 headers to the segmented portions of the stream at step 608, and outputs, at step 610, flows of data packets that  
10 correspond to the segmented portions of the data stream via the appropriate layer 4 port. At this point, the I/O element requests an additional pop of a portion of the data within the stream queue at step 612. After receiving the additional pop from the stream queue at step 614, the I/O element returns to step 606 within the procedure described above.

15

Within Figure 7, the result of the I/O element consuming the data stream is the outputting of data packets 712 to the packet switched network.

Figure 11 illustrates the block diagram of Figure 6 with the signalling depicted for a scenario in which encrypted data is detected within the data stream by the content processing element 80. In this case, the operation of the stream processing node 50 is identical to that described above for Figure 7 until the content processing element has completed its analysis of the copied portion of the stream queue SQ2. In this case, the content processing element 80  
20 determines that the data within the stream is encrypted and, as indicated by signal 714, requests rescheduling of the stream queue SQ2 to the first application processing element 90, which in this scenario has a decryption algorithm and therefore is a decryption processing element. The stream queue controller 72 then modifies the consumer attributes for the stream queue SQ2 such that the  
25 consumer is set as the application processing element 90 and initiates a pop of data from the stream queue SQ2 to the processing element 90 as indicated by  
30

signal 716.

Figure 12 illustrates a flow chart of a set of steps performed by a decryption processing element within a stream processing node according to the present invention. Firstly, the decryption processing element receives a pop of a portion of the data within a stream queue at step 616. Next, the decryption processing element performs decryption procedures on this data at step 618 and pushes the decrypted substreams of a new data stream to a second stream queue within the stream fabric at step 620. Initially, the decryption processing element likely would need to initiate the second stream queue as indicated in the description of Figure 1. Finally, the decryption processing element requests an additional pop of a portion of the data within the first stream queue at step 622 and then proceeds to return to step 616 described herein above.

Turning to Figure 11, the application processing element 90, acting as a decryption processing element, receives portions of the data within stream queue SQ2 and outputs decrypted versions of the data stream into stream queue SQ3 as indicated by signal 718. In this scenario, the application processing element 90 initiated the stream queue SQ3 such that the initial consumer was chosen to be the content processing element 80. In this case, as described previously, a copy of the data within the stream queue SQ3 is copied to the content processing element 80 as indicated by signal 720. After analysing this data, the content processing element 80 then sends instructions to the stream queue controller by signal 714. In the scenario depicted in Figure 11, the content processing element 80 determines the decrypted data stream includes a URL that should be used to properly route the data stream similar to the scenario of Figure 7. This results in the rescheduling of the stream queue SQ3 to the I/O element 60, the popping of portions of the data within the stream queue SQ3 to the I/O element 60 as indicated by signal 722 and the eventual outputting of packets 724 corresponding to the decrypted data stream to the packet switched network.

It should be understood that in an alternative scenario, the content processing element 80, after determining the URL of the decrypted data stream, could have requested that the data within the stream queue SQ3 be re-encrypted prior to being rescheduled to the I/O element 60 for outputting. This could have been accomplished by rescheduling the stream queue SQ3 to an encryption processing element and initiating a third stream queue with the rights of the consumer and producer of this new stream queue being the I/O element 60 and the encryption processing element respectively. It should be understood that other techniques for re-encrypting the data stream could also be utilized.

Although the present invention is primarily described herein above for an implementation within a packet switched network, it should be understood that the scope of the present invention should not be limited to this implementation. It should be noted that the present invention could be used as the basis for switches, routers, bridges, servers, firewalls, load balancers, cache farms, etc. For instance, Figure 13 illustrates a block diagram of an alternative stream processing node to that illustrated in Figure 6 in which the I/O element 60 is replaced by a memory cache 150. In this case, the memory cache 150 could be utilized as a producer and/or a consumer for a stream queue within the stream fabric 70. It should be understood that a plurality of memory cache could be implemented in a stream processing node according to the present invention, possibly with no other processing elements but the content processing element 80.

It should be understood that embodiments of the stream processing node according to the present invention as described above could be implemented within one or more separate computer chips. For instance, in one implementation, an I/O element, a stream fabric and a plurality of processing elements are implemented on a single chip while in other implementations all of or at least a portion of the I/O element is implemented on a separate computer

chip. Further implementations could also be possible.

Persons skilled in the art will appreciate that there are alternative implementations and modifications of the present invention, and that the above  
5 described implementation is only an illustration of specific embodiments of the invention. Therefore, the scope of the invention should only be limited by the claims appended hereto.

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215